基于 Pushlet 的 RFID 数据推送技术研究

梁昌勇,张怡远,张俊岭
(合肥工业大学,安徽 合肥 230009)

摘 要:Pushlet 是一种基于 Servlet 的技术,用来将服务器数据主动推送到客户端,具有构建简单、使用方便、系统环境要求低等优点。RFID 系统因为其自身的技术原因,数据主动推送系统必不可少。RFID 数据具有突发性、间断性、数据量小等特点,特别适合 Pushlet 的应用。以 RFID 数据主动推送系统为例,系统地介绍了 Pushlet 的使用步骤,实现了 RFID 数据主动推送系统,具有高效性、易扩展性和平台无关性,并对 Pushlet 的性能进行了分析。

关键词:Pushlet;服务器推送;RFID;Comet
中图分类号:TP311.5 **文献标识码**:A **文章编号**:1673-629X(2009)10-0085-04

Research on RFID Data Push Technology Based on Pushlet

LIANG Chang-yong, ZHANG Yi-yuan, ZHANG Jun-ling
(Hefei University of Technology, Hefei 230009, China)

Abstract: Pushlet is a technology based on Servlet, which is used to push server data to clients. It has the advantage of simply construct, easy to use and a low requirement of system environment. Data push system is necessary for RFID system, because of its own technical reasons, RFID data has the characteristic of suddenness, discontinuity and small amount of data, which is very suitable for Pushlet. Introduce a procedure of using Pushlet systematically based on an example of RFID data pushing system, realize the RFID data push system with high efficiency, easy scalability and platform independence, then analyze the performance of Pushlet framework.

Key words: Pushlet; server push; RFID; Comet

0 引 言

射频识别技术(RFID)是20世纪90年代兴起的一种自动识别技术,是射频技术在识别领域中的应用。RFID读写器通过非接触方式读取标签数据,完成系统基础数据的自动采集工作,成为快速准确获取原始基础数据的有效工具。将RFID技术应用在汽车制造过程控制领域可以提高工作效率,减少安装错误率,提供完整的零部件质量信息。

系统构建在网络环境下,通过服务器主动推送方式,将采集到的RFID数据实时准确方便地从服务器推送给客户端。文中基于Pushlet框架,介绍一种轻量、易实现、系统要求低的数据推送解决方案。

1 RFID 系统结构

RFID系统由标签、读写器、采集设备、处理设备组成。当标签进入读写器场强范围内,依靠场强提供的能量激活标签自身芯片,将标签中存储的卡号信息发送给读写器。与传统的条码、磁卡和IC卡相比,RFID标签具有非接触、读写快、寿命长、不怕污染和同时处理多张标签等优点。RFID为管理信息系统提供实时准确物流信息,是实体世界与计算机虚拟世界沟通的桥梁。

随着网络技术的发展,RFID系统越来越多地构建在网络环境下。这一趋势造成了数据采集设备与数据处理设备的分离;而且可能出现一台读写器读取的标签数据被多个客户端调用,或者多台读写器组成读写器网络读取同一个标签的现象;同时数据采集设备不能确定什么时候能读到标签,所以RFID系统不适合采用传统的"请求/响应"(Request/Response)方式,需要一个数据主动推送的方案,将RFID数据动态实时地从采集设备推送给处理设备。如图1所示。

2 在网络环境下基于 Pushlet 推送技术的 RFID 应用系统分析与设计

RFID应用在汽车制造过程控制领域,由于RFID技术的自身特点和汽车制造企业的实际情况,整个系

收稿日期:2009-01-12;**修回日期**:2009-04-23
基金项目:国家863项目(2006AA04A126)
作者简介:梁昌勇(1965-),男,安徽肥西人,博士,教授,博士生导师,研究方向为RFID系统集成、RFID体系架构。

统采用 B/S(Browser/Server)架构,分为数据采集、数据清洗、数据融合、数据发送、客户端业务处理等几个模块[1]。

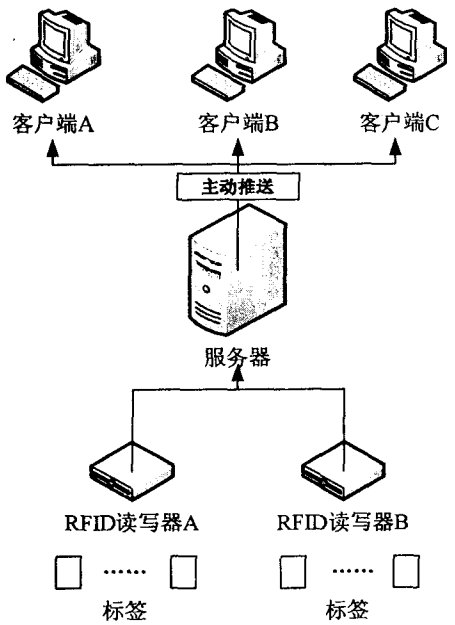

图 1 网络环境下 RFID 系统结构图

数据采集模块负责控制多台 RFID 读写器,当一台或多台读写器读取到标签后,将标签数据发往服务器。数据清洗模块对采集到的标签数据进行预处理,删除重复读取和无效读取数据。数据融合模块将多个读写器读到的数据加以融合,获得某个标签的一次有效读取数据。数据推送模块将该有效读取数据推送给一台或多台客户端[2]。最后客户端根据实际业务需要对接受到的 RFID 数据进行处理。系统结构如图 2 所示。

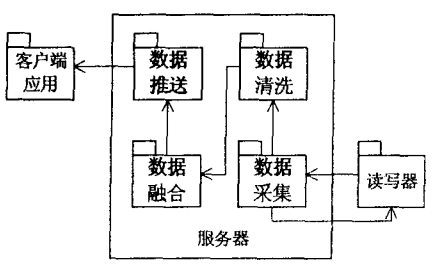

图 2 系统模块图

现有的数据推送方案主要有套接口通讯、对象回调、消息服务等方式。但这几种方式各有不足,例如套接口通讯因为使用非 HTTP 端口容易被防火墙阻拦[3];对象回调实现起来比较复杂[4];消息服务需要应用服务器的支持,而且资源消耗大[5]。根据 RFID 系统自身的特点,例如标签数据量小,读取时间没有规律性,延时要求较高,以上几种方案都不是很理想。Pushlet 是一个轻量的、开源的、Comet 的实现框架[6],非常方便地提供了服务器端数据向客户端推送的功能。它集成了上述几种方法的优点,构建简单,系统资源占用低,系统环境要求低,延时较小,特别适用于 RFID 数据推送领域。其工作原理是在服务器端使用一个 Servlet,在客户端发送请求(Request)后,通过标准的 HTTP 端口与客户端建立连接,将需发送数据放到响应(Response)中推送给客户端,推送完成后该连接并不断开,继续等待新的数据并推送。Pushlet 在客户端提供了 Java 类和动态 HTML 两种形式接收数据[7]。

Pushlet 连接方式如图 3 所示。

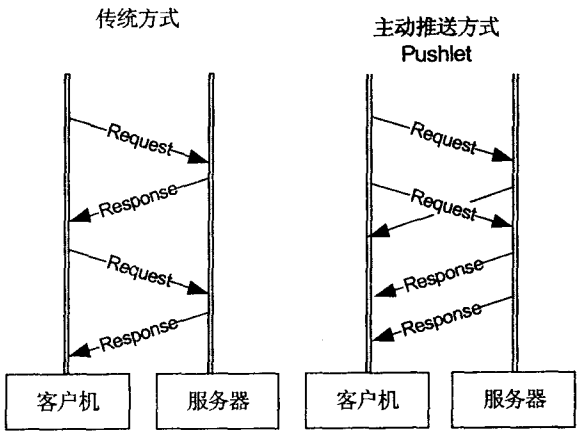

图 3 Pushlet 连接示意图

Pushlet 无需任何 Applet 或者插件的帮助,因而可以在任何支持 Servlet 的服务器中部署(如 Tomcat)。同时对浏览器的要求也很低,只要是兼容 JavaScript1.4 版本以上的浏览器(如 Internet Explorer、FireFox)即可[8]。

Pushlet 通过主题来管理所要推送的数据,并采用具有层次的树形结构。上层主题包含下层主题,默认以"/"作为分隔符。例如:"/"表示所有事件,"/RFID"表示所有 RIFD 事件,"/RFID/Device1"表示第一台 RFID 设备产生的事件,"/RFID/Device2"表示第二台 RFID 设备产生的事件。如果客户端订阅的是"/RFID",则可以收到两个 RFID 设备所产生的事件[9]。

3 应用系统实现

3.1 服务器端原理与实现

Pushlet 框架服务器端获取到信息后,将要推送的信息打包成一个事件(Event)对象,并指明这个事件的主题,然后调用调度器(Dispatcher)的 multicast 方法进行推送。调度器会查询内部的客户端订阅列表。当有客户端订阅的主题和当前事件主题相同时,Pushlet 通过调用各个订阅者的 send 方法,把事件对象推送到客户端。

(1)新建一个 Java 类 RFID,构建一个内部类

RFIDTag,用来封装要推送的消息[10]。

```
class RFIDTag {
public String tag = "EMPTY"; //RFID 标签信息
public String time = "EMPTY"; //标签读取时间
public boolean modified = false;}//当前对象是否需要被推送
```

(2)构建一个数据集合 cache,用来暂存推送的数据。

```
Vector<RFIDTag> cache = new Vector<RFIDTag> (Default_Length);
```

(3)再构建一个静态内部类 RFIDReaderEventPush,作为推送资源,并继承 EventSource 和 Runnable 接口。

```
static public class RFIDReaderEventPush implements EventSource, Runnable{}
```

(4)实现 EventSource 和 Runnable 接口中的 4 个方法,分别是 active(新建线程,激活当前资源),Stop(结束这个资源),passivate(挂起当前资源),run(发布资源)。

```
public void run() {
while (active) {
updateCache();//更新 cache,本例中就是调用 RFID 读写器获取标签信息
publish();} }//推送数据
//publish()只会推送 RFIDTag 的 modified 字段为 True 的对象
private void publish () {
for (int i = 0; i < cache.size(); i++) {
RFIDTag nextTag = (RFIDTag) cache.elementAt(i);
if (nextTag.modified) {
publishTag(i, nextTag.tag,nextTag.time);//实际推送
nextTag.modified = false;}}}//推送完成后将 modified 标记置为假
```

(5)实现 publishTag 方法。

```
private void publishTag (int index, String tag, String time) {
Event event = Event.createDataEvent("/RFID");//创建一个主题为"/RFID"的事件
event.setField("tag",tag); //要推送的信息放到事件里面去
event.setField("time",time);
Dispatcher.getInstance().multicast(event);}//推送该事件
```

(6)发布 Pushlet 资源。

修改在网站发布目录 WEB-INF\classes 文件夹下的 sources.properties 文件。

```
Source100 = nl.justobjects.pushlet.test.RFID $ RFIDReaderEventPush
```

3.2 Java 类客户端原理与实现

客户端通过建立一个 PushletClient 对象,在指明要订阅的主题和推送模式后,调用 joinListen 方法,将自己加入到 Pushlet 内部的订阅者列表。同时客户端需要实现 PushletClientListener 接口中的 onData 方法,指明接收到推送数据后的处理方法,通过 Event 的 getField 方法获得封装在 Event 中的数据。Java 类形式的客户端,其表示层既可以是 Web 页面,也可以是应用程序(Application)。具体的应用步骤如下:

(1)客户端新建一个类 RFIDClient,并继承 PushletClientListener 和 Protocol 接口。

```
public class RFIDClient implements PushletClientListener, Protocol {}
```

(2)编写 RFIDClient 类的 Main 方法。

```
PushletClient pushletClient = new PushletClient(aHost,aPort);
pushletClient.join();
String SUBJECT = "/RFID";//推送主题
final String MODE = MODE_STREAM;//推送模式
pushletClient.joinListen(this, MODE, SUBJECT);//客户端加入该主题的监听列表
```

(3)实现 PushletClientListener 的 4 个方法,分别是 onData(接收到推送数据时),onHeartbeat(接收到服务器心跳信息),onError(出现错误时),onAbort(被中止时)。Pushlet 在 Protocol 接口中定义了很多常量,没有定义方法。

```
public void onData(Event theEvent) {
System.out.println ("tag:" + theEvent.getField("tag"));//通过 getField 方法,获取推送数据
System.out.println ("time:" + theEvent.getField("time"));}
```

3.3 动态 HTML 客户端原理与实现

动态 HTML 形式的客户端实现原理和 Java 类形式大体相同,区别在于这种形式客户端直接嵌入在网页中[11]。通过 JavaScript 中对主题进行订阅,并提供接受到信息后的处理方法。这种方式十分简单,体现了 Pushlet 作为轻量级框架的特点。

(1)加入相关的 JavaScript 脚本。

```
<script type = "text/javascript" src = "../../lib/js - pushlet - client.js"></script>
<script type = "text/javascript">p_embed()</script>
```

(2)创建一个文本域用来显示推送的数据。

```
<textarea cols = "15" rows = "12" name = 'EVENT">
```

(3)在 Javascript 中创建两个方法,订阅主题和获取推送事件。

```
<script type = "text/javascript">
function init() {
p_join_listen('/RFID');}//订阅一个主题
function onEvent(event) {//收到推送事件后的处理方法
document.eventDisplay.EVENT.value = event.toString();} </script>
```

(4)在页面加载是时候初始化 Pushlet,调用 init() 函数。

```
<body onLoad = "init()">
```

4 Pushlet 性能分析

原型系统采用一台 Intel Core E8200,2G 内存服务器;六台 P4 2.4G,1.5G 内存的客户机,网络采用 10M 全双工模式经行互联。分别对消息完整性、响应时间、多用户大负荷情况等几个方面进行了测试。

在实验中服务器以一个 50～500 毫秒的随机时间间隔,发送一个包括自增长序列号和当前服务器时间的数据包。客户机收到消息后,首先计算出一共收到多少条消息,再与自增长序列号比较,其次用客户机当前时间与数据包中的服务器时间进行比较。

在网络轻负荷情况下,客户机接收到推送信息平均延时的分布直方图如图 4 所示。平均延时大概为 450 毫秒左右。

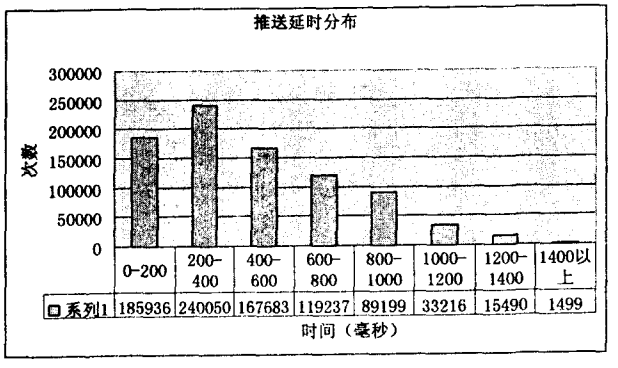

图 4 延时分布直方图

系统资源占用方面,以常见的消息方式(JMS)进行对比。因为 Tomcat 不支持 JMS,所以选用 Jboss 为 JMS 的应用服务器。同等的推送强度下,JMS 方式的 CPU 和内存占用量约为 Pushlet 方式的 2～5 倍。

Pushlet 稳定性很高,在多用户多主题的情况下,服务器发送了 1000 万条测试数据后,都没有出现丢失和重复发送的现象。

在实验中,基于 Pushlet 的主动推送框架还表现出以下优点:直接与动态 HTML 集成;构建简单,只需要使用支持 Servlet 的服务器即可;使用标准 HTTP 端口进行连接,不会被防火墙拦截。

5 结束语

文中分析在网络环境下,RFID 应用系统的数据推送需求。提出了一种基于 Pushlet 的数据推送框架,并给出了系统的具体实现。最后通过原型系统验证了 Pushlet 框架的应用可行性,具有一定的推广价值。

参考文献:

[1] 邓海生.基于 RFID 的数据采集中间件[J].计算机技术与发展,2007,17(9):188－191.

[2] 邓海生.RFID 中间件研究与设计[J].计算机技术与发展,2008,18(11):55－57.

[3] JavaWorld. An in－depth look at RMI callbacks[EB/OL]. 1999－04－20. http://www.javaworld.com/javaworld/javaqa/1999－04/05－rmicallback.html.

[4] Adobe. ActionScript 2.0 语言参考[EB/OL]. 2003－04. http://livedocs.adobe.com/flash/8-cn/main/00002905.html.

[5] JMS 简介[EB/OL]. 2007－07－14. http://java.csdn.net/page/9cd3a678－9787－4ccc－92be－6fdfd23804fa.

[6] van den Broecke J. Pushlet Cookbook[EB/OL]. 2007－11－24. http://www.pushlets.com/doc/cookbook.html.

[7] Alinone A. Comet and Push Technology[EB/OL]. 2007－10－19. http://cometdaily.com/2007/10/19/comet－and－push－technology.

[8] 周 婷. Comet:基于 HTTP 长连接的"服务器推"技术[EB/OL]. 2007－08－31. http://www.ibm.com/developerworks/cn/web/wa－lo－comet.

[9] Wilkins G. Comet is Always Better Than Polling[EB/OL]. 2007－11－06. http://cometdaily.com/2007/11/06/comet－is－always－better－than－polling.

[10] Matrix. Think in Pushlet[EB/OL]. 2007－01－16. http://www.matrix.org.cn/resource/article/2007－01－16/.html.

[11] Goodman D. Dynamic HTML: The Definitive Reference[M]. 3rd edition. [s.l.]:O'Reilly,2006.

(上接第 84 页)

感技术学报,2005,18(2): 307－312.

[4] 崔 莉,鞠海玲,苗 勇,等. 无线传感器网络研究进展[J].计算机研究与发展,2005,42(1):163－174.

[5] 李 莉,刘元安,唐碧华.一种基于网格模型的传感器节点放置算法[J].武汉大学学报:理学版,2007(S):83－87.

[6] 李 冰,李 捷.一种基于 GAF 的无线传感器网络分簇算法[J].计算机技术与发展,2008,18(12):113－115.

[7] Zou Y,Chakrabarty K. Sensor deployment and target localization based on virtual forces[C]//Proc. IEEE INFOCOM Conference. USA:[s.n.],2003:1293－1303.

[8] Fonseca C M,Fleming P J. Genetic Algorithms for Multi objective Optimization: Formulation, Discussion and Generalization[C]//in Genetic Algorithms. Proc. Fifth International Conference. [s.l.]:Morgan Kaufmann,1993:416－423.

[9] Zhang H, Hou J C. Maintaining Sensing Coverage and Connectivity in Large Sensor Networks[R]. USA: University of Illinois at Urbana－Champaign, 2003.

[10] 曾建潮,崔志华.微粒群算法[M].北京:科学出版社,2000.

[11] Chakrabarty K, Iyengar S S, Qi H, et al. Grid coverage for surveillance and target location in distributed sensor networks [J]. IEEE Transactions on Computers, 2002, 51: 1448－1453.